Two-stage Methods for Linear Discriminant Analysis: Equivalent Results at a Lower Cost*

Peg Howland† and Haesun Park‡

Abstract

Linear discriminant analysis (LDA) has been used for decades to extract features that preserve class separability. It is classically defined as an optimization problem involving covariance matrices that represent the scatter within and between clusters. The requirement that one of these matrices be nonsingular restricts its application to data sets in which the dimension of the data does not exceed the sample size. Recently, the applicability of LDA has been extended by using the generalized singular value decomposition (GSVD) to circumvent the nonsingularity requirement. Alternatively, many studies have taken a two-stage approach in which the first stage reduces the dimension of the data enough so that it can be followed by classical LDA. In this paper, we justify a two-stage approach that uses either principal component analysis or latent semantic indexing before the LDA/GSVD method. We show that it is equivalent to single-stage LDA/GSVD. We also present a computationally simpler choice for the first stage, and conclude with a discussion of the relative merits of each approach.

1 Introduction

The goal of linear discriminant analysis (LDA) is to combine features of the original data in a way that most effectively discriminates between classes. With an appropriate extension, it can be applied to the goal of reducing the dimension of a data matrix in a way that most effectively preserves its cluster structure. That is, we want to find a linear transformation G^T that maps an m-dimensional data vector a to a vector y in the l-dimensional space ($l \ll m$):

$$G^T : a \in \mathbb{R}^{m\times 1} \rightarrow y \in \mathbb{R}^{l\times 1}.$$

Assuming that the given data are already clustered, we seek a transformation that optimally preserves this cluster structure in the reduced dimensional space.

*This work was supported in part by the National Science Foundation grant CCF-0621889. Any opinions, findings and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation (NSF).

†Department of Mathematics and Statistics, Utah State University, 3900 Old Main Hill, Logan, UT 84322, peg.howland@usu.edu

‡College of Computing, Georgia Institute of Technology, hpark@cc.gatech.edu

For simplicity of discussion, we will assume that the m-dimensional data vectors $a_1, \ldots, a_n$ form columns of a matrix $A \in \mathbb{R}^{m \times n}$, and are grouped into k clusters as

$$A = [A_1, A_2, \cdots, A_k] \text{ where } A_i \in \mathbb{R}^{m \times n_i}, \text{ and } \sum_{i=1}^{k} n_i = n. \tag{1}$$

Let N_i denote the set of column indices that belong to cluster i. The centroid $c^{(i)}$ is computed by taking the average of the columns in cluster i; i.e.,

$$c^{(i)} = \frac{1}{n_i} \sum_{j \in N_i} a_j \tag{2}$$

and the global centroid c is defined as

$$c = \frac{1}{n} \sum_{j=1}^{n} a_j. \tag{3}$$

Then the within-cluster, between-cluster, and mixture scatter matrices are defined [Fuk90, TK99] as

$$\begin{aligned} S_w &= \sum_{i=1}^{k} \sum_{j \in N_i} (a_j - c^{(i)})(a_j - c^{(i)})^T, \\ S_b &= \sum_{i=1}^{k} \sum_{j \in N_i} (c^{(i)} - c)(c^{(i)} - c)^T \\ &= \sum_{i=1}^{k} n_i (c^{(i)} - c)(c^{(i)} - c)^T, \text{ and} \\ S_m &= \sum_{j=1}^{n} (a_j - c)(a_j - c)^T, \end{aligned}$$

respectively. The scatter matrices have the relationship [JD88]

$$S_m = S_w + S_b. \tag{4}$$

Applying G^T to the matrix A transforms the scatter matrices S_w, S_b, and S_m to the $l \times l$ matrices

$$G^T S_w G, \quad G^T S_b G, \quad \text{and} \quad G^T S_m G,$$

respectively.

There are several measures of cluster quality that involve the three scatter matrices [Fuk90, TK99]. When cluster quality is high, each cluster is tightly grouped, but well separated from the other clusters. Since

$$\begin{aligned} \text{trace}(S_w) &= \sum_{i=1}^{k} \sum_{j \in N_i} (a_j - c^{(i)})^T (a_j - c^{(i)}) \\ &= \sum_{i=1}^{k} \sum_{j \in N_i} \|a_j - c^{(i)}\|_2^2 \end{aligned}$$

measures the closeness of the columns within the clusters, and

$$\begin{aligned} \mathrm{trace}(S_b) &= \sum_{i=1}^{k} \sum_{j \in N_i} (c^{(i)} - c)^T (c^{(i)} - c) \\ &= \sum_{i=1}^{k} \sum_{j \in N_i} \|c^{(i)} - c\|_2^2 \end{aligned}$$

measures the separation between clusters, an optimal transformation that preserves the given cluster structure would maximize $\mathrm{trace}(G^T S_b G)$ and minimize $\mathrm{trace}(G^T S_w G)$.

This simultaneous optimization can be approximated by finding a transformation G that maximizes

$$J_1(G) = \mathrm{trace}((G^T S_w G)^{-1} G^T S_b G). \tag{5}$$

However, this criterion cannot be applied when the matrix S_w is singular, a situation that occurs frequently in many applications. For example, in handling document data in information retrieval, it is often the case that the number of terms in the document collection is larger than the total number of documents (i.e., $m > n$ in the term-document matrix A), and therefore the matrix S_w is singular. Furthermore, for applications where the data points are in a very high-dimensional space rendering data collection expensive, S_w is singular because the value for n must be kept relatively small. Such is the case for the image databases of face recognition, as well as for gene expression data. This is referred to as the small sample size problem, or the problem of undersampled data.

Classical LDA expresses the solution in terms of an eigenvalue problem when S_w is nonsingular. By reformulating the problem in terms of the generalized singular value decomposition (GSVD) [VL76, PS81, GVL96], the LDA/GSVD algorithm [HJP03] extends the applicability to the case when S_w is singular. Another way to apply LDA to the data matrix $A \in \mathbb{R}^{m \times n}$ with $m > n$ (and hence S_w singular) is to perform dimension reduction in two stages. The LDA stage is preceded by a stage in which the cluster structure is ignored. A common approach for the first part of this process is rank reduction by the truncated singular value decomposition (SVD). This is the main tool in principal component analysis (PCA) [DHS01], as well as in latent semantic indexing (LSI) [DDF$^+$90, BDO95] of documents. Swets and Weng [SW96] and Belhumeur et al. [BHK97] utilized PCA plus LDA for facial feature extraction. Torkkola [Tor01] implemented LSI plus LDA for document classification. A drawback of these two-stage approaches has been the experimentation needed to determine which intermediate reduced dimension produces optimal results after the second stage.

Moreover, since either PCA or LSI ignores the cluster structure, theoretical justification for such two-stage approaches has been lacking. Yang and Yang [YY03] justify PCA plus LDA, providing proof in terms of a single discriminant vector. They do not address the optimal reduced dimension after the second stage. In this paper, we establish the equivalence of single-stage LDA/GSVD to two-stage methods consisting of either PCA or LSI, followed by LDA/GSVD. In the range of intermediate dimensions for which equivalence is achieved, S_w remains singular, and hence classical LDA cannot be used for the second stage. An implication of this equivalence is that the optimal set of discriminant vectors after the second stage should include the same number of vectors as LDA when used alone. With the GSVD framework, we clearly show that at most $k - 1$ generalized eigenvectors are needed even in the singular case. Thus, in addition to its role

in the LDA/GSVD algorithm, the GSVD provides a mathematical framework for understanding the singular case, and eliminates the need for experimentation to determine the optimal reduced dimension.

We also present a computationally simpler choice for the first stage, which uses QR decomposition (QRD) rather than the SVD. After confirming the equivalence of these approaches experimentally, we discuss the relative merits of each. We conclude that QRD plus LDA, which uses QRD as a pre-processing step for LDA/GSVD, provides an improved algorithm for LDA/GSVD.

2 LDA based on the GSVD

It is well-known that the J_1 criterion (5) is maximized when the columns of G are the l eigenvectors of $S_w^{-1}S_b$ corresponding to the l largest eigenvalues [Fuk90]. In other words, classical discriminant analysis solves

$$S_w^{-1}S_b x_i = \lambda_i x_i \tag{6}$$

for the x_i's corresponding to the largest λ_i's. For these l eigenvectors, the maximum achieved is $J_1(G) = \lambda_1 + \cdots + \lambda_l$. Furthermore, $\mathrm{rank}(S_b)$ of the eigenvalues of $S_w^{-1}S_b$ are greater than zero, and the rest are zero. Hence, for $l \geq \mathrm{rank}(S_b)$, this optimal G preserves $\mathrm{trace}(S_w^{-1}S_b)$ exactly upon dimension reduction.

In terms of the data clusters and centroids given in (1), (2), and (3), [HJP03] defines the $m \times n$ matrices

$$H_w = [A_1 - c^{(1)}e^{(1)^T}, A_2 - c^{(2)}e^{(2)^T}, \ldots, A_k - c^{(k)}e^{(k)^T}] \tag{7}$$

$$H_b = [(c^{(1)} - c)e^{(1)^T}, (c^{(2)} - c)e^{(2)^T}, \ldots, (c^{(k)} - c)e^{(k)^T}]$$

$$H_m = [a_1 - c, \ldots, a_n - c] = A - ce^T, \tag{8}$$

where $e^{(i)} = (1, \ldots, 1)^T \in \mathbb{R}^{n_i \times 1}$ and $e = (1, \cdots, 1)^T \in \mathbb{R}^{n \times 1}$. Then the scatter matrices can be expressed as

$$S_w = H_w H_w^T, \quad S_b = H_b H_b^T, \quad \text{and} \quad S_m = H_m H_m^T. \tag{9}$$

Another way to define H_b that satisfies (9) is

$$H_b = [\sqrt{n_1}(c^{(1)} - c), \sqrt{n_2}(c^{(2)} - c), \ldots, \sqrt{n_k}(c^{(k)} - c)] \tag{10}$$

and using this $m \times k$ form reduces the storage requirements and computational complexity of the LDA/GSVD algorithm.

As the product of an $m \times n$ matrix and an $n \times m$ matrix, S_w is singular when $m > n$ [Ort87]. This means that J_1 cannot be applied when the number of available data points is smaller than the dimension of the data. Expressing λ_i as α_i^2/β_i^2, the eigenvalue problem (6) becomes

$$\beta_i^2 H_b H_b^T x_i = \alpha_i^2 H_w H_w^T x_i. \tag{11}$$

This has the form of a problem that can be solved using the GSVD of the matrix pair (H_b^T, H_w^T). Paige and Saunders [PS81] defined the GSVD for any two matrices with the same number of columns, which we restate as follows.

Theorem 2.1 *Suppose two matrices $H_b^T \in \mathbb{R}^{k\times m}$ and $H_w^T \in \mathbb{R}^{n\times m}$ are given. Then for*

$$K = \begin{pmatrix} H_b^T \\ H_w^T \end{pmatrix} \quad \textit{and} \quad t = rank(K),$$

there exist orthogonal matrices $U \in \mathbb{R}^{k\times k}$, $V \in \mathbb{R}^{n\times n}$, $W \in \mathbb{R}^{t\times t}$, and $Q \in \mathbb{R}^{m\times m}$ such that

$$U^T H_b^T Q = \Sigma_b(\underbrace{W^T R}_{t}, \underbrace{0}_{m-t})$$

and

$$V^T H_w^T Q = \Sigma_w(\underbrace{W^T R}_{t}, \underbrace{0}_{m-t}),$$

where

$$\underset{k\times t}{\Sigma_b} = \begin{pmatrix} I_b & & \\ & D_b & \\ & & O_b \end{pmatrix}, \quad \underset{n\times t}{\Sigma_w} = \begin{pmatrix} O_w & & \\ & D_w & \\ & & I_w \end{pmatrix},$$

and $R \in \mathbb{R}^{t\times t}$ is nonsingular with its singular values equal to the nonzero singular values of K. The matrices

$$I_b \in \mathbb{R}^{r\times r} \quad \textit{and} \quad I_w \in \mathbb{R}^{(t-r-s)\times(t-r-s)}$$

are identity matrices, where

$$r = t - rank(H_w^T) \quad \textit{and} \quad s = rank(H_b^T) + rank(H_w^T) - t,$$

$$O_b \in \mathbb{R}^{(k-r-s)\times(t-r-s)} \quad \textit{and} \quad O_w \in \mathbb{R}^{(n-t+r)\times r}$$

are zero matrices with possibly no rows or no columns, and

$$D_b = diag(\alpha_{r+1}, \ldots, \alpha_{r+s})$$

and

$$D_w = diag(\beta_{r+1}, \ldots, \beta_{r+s})$$

satisfy

$$1 > \alpha_{r+1} \geq \cdots \geq \alpha_{r+s} > 0, \quad 0 < \beta_{r+1} \leq \cdots \leq \beta_{r+s} < 1,$$

and $\alpha_i^2 + \beta_i^2 = 1$ for $i = r+1, \ldots, r+s$.

This form of GSVD is related to that of Van Loan [VL76] as

$$U^T H_b^T X = (\Sigma_b, 0) \quad \text{and} \quad V^T H_w^T X = (\Sigma_w, 0), \tag{12}$$

where

$$\underset{m\times m}{X} = Q \begin{pmatrix} R^{-1}W & 0 \\ 0 & I_{m-t} \end{pmatrix}.$$

This implies that

$$X^T H_b H_b^T X = \begin{pmatrix} \Sigma_b^T \Sigma_b & 0 \\ 0 & 0 \end{pmatrix} \quad \text{and} \quad X^T H_w H_w^T X = \begin{pmatrix} \Sigma_w^T \Sigma_w & 0 \\ 0 & 0 \end{pmatrix}. \tag{13}$$

Letting x_i represent the ith column of X, and defining

$$\alpha_i = 1,\ \beta_i = 0 \text{ for } i = 1, \ldots, r$$

and

$$\alpha_i = 0,\ \beta_i = 1 \text{ for } i = r + s + 1, \ldots, t,$$

we see from (13) that (11) is satisfied for $1 \leq i \leq t$. Since

$$H_b H_b^T x_i = 0 \quad \text{and} \quad H_w H_w^T x_i = 0$$

for the remaining $m - t$ columns of X, (11) is satisfied for arbitrary values of α_i and β_i when $t + 1 \leq i \leq m$. The columns of X are the generalized singular vectors for the matrix pair (H_b^T, H_w^T). They correspond to the generalized singular values, or the α_i/β_i quotients, as follows. The first r columns correspond to infinite values, and the next s columns correspond to finite and nonzero values. The following $t - r - s$ columns correspond to zero values, and the last $m - t$ columns correspond to the arbitrary values. This correspondence between generalized singular vectors and values is summarized in Table 1.

Table 1: Generalized singular vectors, $\{x_i : 1 \leq i \leq m\}$, and their corresponding generalized singular values, $\{\alpha_i/\beta_i : 1 \leq i \leq m\}$

i	α_i/β_i	$x_i \in \text{null}(S_w)$?	$x_i \in \text{null}(S_b)$?
$1, \ldots, r$	∞	yes	no
$r + 1, \ldots, r + s$	positive	no	no
$r + s + 1, \ldots, t$	0	no	yes
$t + 1, \ldots, m$	arbitrary	yes	yes

A question that remains is which columns of X to include in the solution G. If S_w is nonsingular, both $r = 0$ and $m - t = 0$, so $s = \text{rank}(H_b^T)$ generalized singular values are finite and nonzero, and the rest are zero. Hence we include in G the leftmost s columns of X. For the case when S_w is singular, [HJP03] argues in terms of the simultaneous optimization

$$\max_G \text{trace}(G^T S_b G) \text{ and } \min_G \text{trace}(G^T S_w G) \tag{14}$$

that criterion J_1 is approximating. Letting g_j represent a column of G, we write

$$\text{trace}(G^T S_b G) = \sum g_j^T S_b g_j$$

and

$$\text{trace}(G^T S_w G) = \sum g_j^T S_w g_j.$$

If x_i is one of the leftmost r vectors, then $x_i \in \text{null}(S_w) - \text{null}(S_b)$. Because $x_i^T S_b x_i > 0$ and $x_i^T S_w x_i = 0$, including this vector in G increases the trace we want to maximize while leaving the trace we want to minimize unchanged. On the other hand, for the rightmost $m - t$ vectors, $x_i \in \text{null}(S_w) \cap \text{null}(S_b)$. Adding the column x_i to G has no effect on these traces, since $x_i^T S_w x_i = 0$

and $x_i^T S_b x_i = 0$, and therefore does not contribute to either maximization or minimization in (14). We conclude that, whether S_w is singular or nonsingular, G should be comprised of the leftmost $r + s = \mathrm{rank}(H_b^T)$ columns of X.

As a practical matter, LDA/GSVD includes the first $k-1$ columns of X in G. This is due to the fact that $\mathrm{rank}(H_b) \leq k-1$, which is clear from the definition of H_b given in (10). If $\mathrm{rank}(H_b) < k-1$, including extra columns in G (some which correspond to the $t-r-s$ zero generalized singular values and, possibly, some which correspond to the arbitrary generalized singular values) will have approximately no effect on cluster preservation. As summarized in Algorithm 1, we first compute the matrices H_b and H_w from the data matrix A. We then solve for a very limited portion of the GSVD of the matrix pair (H_b^T, H_w^T). This solution is accomplished by following the construction in the proof of Theorem 2.1 [PS81]. The major steps are limited to the complete orthogonal decomposition [GVL96, LH95] of

$$K = \begin{pmatrix} H_b^T \\ H_w^T \end{pmatrix},$$

which produces orthogonal matrices P and Q and a nonsingular matrix R, followed by the singular value decomposition of a leading principal submatrix of P, whose size is much smaller than that of the data matrix. Finally, we assign the leftmost $k-1$ generalized singular vectors to G. As a result, the linear transformation G^T reduces an m-dimensional data vector a to a vector y of dimension $k-1$.

Another consequence of Theorem 2.1 is that the trace values after LDA/GSVD are on the order of the reduced dimension $k-1$. Due to the normalization $\alpha_i^2 + \beta_i^2 = 1$ for $1 \leq i \leq m$ in the Paige and Saunders formulation, we have

Corollary 2.2 *The LDA/GSVD algorithm reduces* $\mathrm{trace}(S_m)$ *to the number of clusters minus 1.*

Proof. From the expressions (13) for the matrix pair (H_b^T, H_w^T), we have

$$S_b = X^{-T} \begin{pmatrix} \Sigma_b^T \Sigma_b & 0 \\ 0 & 0 \end{pmatrix} X^{-1} \quad \text{and} \quad S_w = X^{-T} \begin{pmatrix} \Sigma_w^T \Sigma_w & 0 \\ 0 & 0 \end{pmatrix} X^{-1}.$$

Since $G = X \begin{pmatrix} I_{k-1} \\ 0 \end{pmatrix}$,

$$\begin{aligned} \mathrm{trace}(G^T S_b G) &= \mathrm{trace}\left((I_{k-1}, 0) \begin{pmatrix} \Sigma_b^T \Sigma_b & 0 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} I_{k-1} \\ 0 \end{pmatrix} \right) \\ &= \alpha_1^2 + \cdots + \alpha_{k-1}^2. \end{aligned}$$

Likewise, $\mathrm{trace}(G^T S_w G) = \beta_1^2 + \cdots + \beta_{k-1}^2$. Using the scatter matrix relationship (4), we have

$$\begin{aligned} \mathrm{trace}(G^T S_m G) &= \mathrm{trace}(G^T S_b G) + \mathrm{trace}(G^T S_w G) \\ &= (\alpha_1^2 + \beta_1^2) + \cdots + (\alpha_{k-1}^2 + \beta_{k-1}^2) \\ &= k-1. \end{aligned}$$

□

3 Rank reduction based on the truncated SVD

As mentioned in the introduction, two-stage approaches to dimension reduction typically use the truncated SVD in the first stage. Either PCA or LSI may be used; they differ only in that PCA centers the data by subtracting the global centroid from each column of A. In this section, we express both methods in terms of the maximization of $J_2(G) = \mathrm{trace}(G^T S_m G)$.

If we let $G \in \mathbb{R}^{m\times l}$ be any matrix with full column rank, then essentially $J_2(G)$ has no upper bound and maximization is meaningless. Now, let us restrict the solution to the case when G has orthonormal columns. Then there exists $G' \in \mathbb{R}^{m\times(m-l)}$ such that $\begin{pmatrix} G, & G' \end{pmatrix}$ is an orthogonal matrix. In addition, since S_m is positive semidefinite, we have

$$\mathrm{trace}(G^T S_m G) \leq \mathrm{trace}(G^T S_m G) + \mathrm{trace}((G')^T S_m G') = \mathrm{trace}(S_m).$$

Reserving the following notation for the SVD of A:

$$A = U\Sigma V^T, \tag{15}$$

let the SVD of H_m be given by

$$H_m = A - ce^T = \tilde{U}\tilde{\Sigma}\tilde{V}^T. \tag{16}$$

where $\tilde{U} \in \mathbb{R}^{m\times m}$ and $\tilde{V} \in \mathbb{R}^{n\times n}$ are orthogonal, $\tilde{\Sigma} = diag(\tilde{\sigma}_1 \cdots \tilde{\sigma}_n) \in \mathbb{R}^{m\times n}$ (provided $m \geq n$), and the singular values are ordered as $\tilde{\sigma}_1 \geq \tilde{\sigma}_2 \geq \cdots \geq \tilde{\sigma}_n \geq 0$ [GVL96, Bjö96]. Then

$$S_m = H_m H_m^T = \tilde{U}\tilde{\Sigma}\tilde{\Sigma}^T\tilde{U}^T.$$

Hence the columns of $\tilde{U}$ form an orthonormal set of eigenvectors of S_m corresponding to the nonincreasing eigenvalues on the diagonal of $\Lambda = \tilde{\Sigma}\tilde{\Sigma}^T = \mathrm{diag}(\tilde{\sigma}_1^2, \ldots, \tilde{\sigma}_n^2, 0, \ldots 0)$.

Proposition 3.1 *PCA to a dimension of at least* $rank(H_m)$ *preserves* $trace(S_m)$.

Proof. For

$$p = \mathrm{rank}(H_m),$$

if we denote the first p columns of $\tilde{U}$ by $\tilde{U}_p$, and let $\Lambda_p = \mathrm{diag}(\tilde{\sigma}_1^2, \ldots, \tilde{\sigma}_p^2)$, we have

$$\begin{aligned} J_2(\tilde{U}_p) &= \mathrm{trace}(\tilde{U}_p^T S_m \tilde{U}_p) \\ &= \mathrm{trace}(\tilde{U}_p^T \tilde{U}_p \Lambda_p) \\ &= \tilde{\sigma}_1^2 + \cdots + \tilde{\sigma}_p^2 \\ &= \mathrm{trace}(S_m). \end{aligned} \tag{17}$$

This means that we preserve $\mathrm{trace}(S_m)$ if we take $\tilde{U}_p$ as G. Clearly, the same is true for $\tilde{U}_l$ with $l \geq p$. □

Proposition 3.2 *LSI to any dimension greater than or equal to* $rank(A)$ *also preserves* $trace(S_m)$.

Proof. Suppose x is an eigenvector of S_m corresponding to the eigenvalue $\lambda \neq 0$. Then

$$S_m x = \sum_{j=1}^{n} (a_j - c)(a_j - c)^T x = \lambda x.$$

This means $x \in \text{span}\{a_j - c | 1 \leq j \leq n\}$, and hence $x \in \text{span}\{a_j | 1 \leq j \leq n\}$. Accordingly,

$$\text{range}(\tilde{U}_p) \subseteq \text{range}(A).$$

From (15), we write

$$A = U_q \Sigma_q V_q^T \quad \text{for} \quad q = \text{rank}(A), \tag{18}$$

where U_q and V_q denote the first q columns of U and V, respectively, and $\Sigma_q = \Sigma(1:q, 1:q)$. Then $\text{range}(A) = \text{range}(U_q)$, which implies that

$$\text{range}(\tilde{U}_p) \subseteq \text{range}(U_q).$$

Hence

$$\tilde{U}_p = U_q W$$

for some matrix $W \in \mathbb{R}^{q \times p}$ with orthonormal columns. This yields

$$\begin{aligned} J_2(\tilde{U}_p) &= J_2(U_q W) \\ &= \text{trace}(W^T U_q^T S_m U_q W) \\ &\leq \text{trace}(U_q^T S_m U_q) \\ &= J_2(U_q), \end{aligned}$$

where the inequality is due to the fact that $U_q^T S_m U_q$ is positive semidefinite. Since $J_2(\tilde{U}_p) = \text{trace}(S_m)$ from (17), we have $J_2(U_q) = \text{trace}(S_m)$ as well. The same argument holds for U_l with $l \geq q$. □

Corollary 3.3 *In the range of reduced dimensions for which PCA and LSI preserve $trace(S_m)$, they preserve $trace(S_w)$ and $trace(S_b)$ as well.*

Proof. This follows from the scatter matrix relationship (4) and the inequalities

$$\begin{aligned} \text{trace}(G^T S_w G) &\leq \text{trace}(S_w) \\ \text{trace}(G^T S_b G) &\leq \text{trace}(S_b), \end{aligned}$$

which are satisfied for any G with orthonormal columns, since S_w and S_b are positive semidefinite. □

In summary, the individual traces of S_m, S_w, and S_b are preserved by using PCA to reduce to a dimension of at least $\text{rank}(H_m)$, or by using LSI to reduce to a dimension of at least $\text{rank}(A)$.

4 LSI Plus LDA

In this section, we establish the equivalence of the LDA/GSVD method to a two-stage approach composed of LSI followed by LDA, and denoted by LSI + LDA. Using the notation (18), the q-dimensional representation of A after the LSI stage is

$$B = U_q^T A,$$

and the second stage applies LDA to B. Letting the superscript B denote matrices after the LSI stage, we have

$$H_b^B = U_q^T H_b \quad \text{and} \quad H_w^B = U_q^T H_w.$$

Hence

$$S_b^B = U_q^T H_b H_b^T U_q \quad \text{and} \quad S_w^B = U_q^T H_w H_w^T U_q.$$

Suppose

$$S_b^B x = \lambda S_w^B x;$$

i.e. x and λ are an eigenvector-eigenvalue pair of the generalized eigenvalue problem that LDA solves in the second stage. Then, for $\lambda = \alpha^2/\beta^2$,

$$\beta^2 U_q^T H_b H_b^T U_q x = \alpha^2 U_q^T H_w H_w^T U_q x.$$

Suppose the matrix $\begin{pmatrix} U_q, & U_q' \end{pmatrix}$ is orthogonal. Then $(U_q')^T A = (U_q')^T U_q \Sigma_q V_q^T = 0$, and accordingly, $(U_q')^T H_b = 0$ and $(U_q')^T H_w = 0$, since the columns of both H_b and H_w are linear combinations of the columns of A. Hence

$$\begin{aligned} \beta^2 \begin{pmatrix} U_q^T \\ (U_q')^T \end{pmatrix} H_b H_b^T U_q x &= \begin{pmatrix} \beta^2 U_q^T H_b H_b^T U_q x \\ 0 \end{pmatrix} \\ &= \begin{pmatrix} \alpha^2 U_q^T H_w H_w^T U_q x \\ 0 \end{pmatrix} \\ &= \alpha^2 \begin{pmatrix} U_q^T \\ (U_q')^T \end{pmatrix} H_w H_w^T U_q x, \end{aligned}$$

which implies

$$\beta^2 H_b H_b^T (U_q x) = \alpha^2 H_w H_w^T (U_q x).$$

That is, $U_q x$ and α/β are a generalized singular vector and value of the generalized singular value problem that LDA solves when applied to A. To show that these $U_q x$ vectors include *all* the LDA solution vectors for A, we show that $\mathrm{rank}(S_m^B) = \mathrm{rank}(S_m)$. From the definition (8), we have

$$H_m = A - ce^T = A(I - \frac{1}{n} ee^T) = U_q \Sigma_q V_q^T (I - \frac{1}{n} ee^T)$$

and by definition

$$H_m^B = U_q^T H_m,$$

and hence

$$H_m = U_q H_m^B.$$

Since H_m and H_m^B have the same null space, their ranks are the same. This means that the number of non-arbitrary generalized singular value pairs is the same for LDA/GSVD applied to B, which produces $t = \mathrm{rank}(S_m^B)$ pairs, and LDA/GSVD applied to A, which produces $t = \mathrm{rank}(S_m)$ pairs.

We have shown the following.

Theorem 4.1 *If G is an optimal LDA transformation for B, which is the q-dimensional representation of the matrix A via LSI, then $U_q G$ is an optimal LDA transformation for A.*

In other words, LDA applied to A produces

$$Y = (U_q G)^T A = G^T U_q^T A = G^T B,$$

which is the same result as applying LSI to reduce the dimension to q, followed by LDA. Finally, we note that if the dimension after the LSI stage is at least $\mathrm{rank}(A)$, that is $B = U_l^T A$ for $l \geq q$, the equivalency argument remains unchanged.

5 PCA Plus LDA

As we did in the previous section for LSI, we now show that a two-stage approach in which PCA is followed by LDA is equivalent to LDA applied directly to A. From (16), we write

$$H_m = \tilde{U}_p \tilde{\Sigma}_p \tilde{V}_p^T \quad \text{for} \quad p = \mathrm{rank}(H_m), \tag{19}$$

where $\tilde{U}_p$ and $\tilde{V}_p$ denote the first p columns of $\tilde{U}$ and $\tilde{V}$, respectively, and $\tilde{\Sigma}_p = \tilde{\Sigma}(1:p, 1:p)$. Then the p-dimensional representation of A after the PCA stage is

$$B = \tilde{U}_p^T A,$$

and the second stage applies LDA/GSVD to B. Letting the superscript B denote matrices after the PCA stage, we have

$$H_b^B = \tilde{U}_p^T H_b \quad \text{and} \quad H_w^B = \tilde{U}_p^T H_w.$$

Hence

$$S_b^B = \tilde{U}_p^T H_b H_b^T \tilde{U}_p \quad \text{and} \quad S_w^B = \tilde{U}_p^T H_w H_w^T \tilde{U}_p.$$

Suppose

$$S_b^B x = \lambda S_w^B x;$$

i.e. x and λ are an eigenvector-eigenvalue pair of the generalized eigenvalue problem that LDA solves in the second stage. Then, for $\lambda = \alpha^2/\beta^2$,

$$\beta^2 \tilde{U}_p^T H_b H_b^T \tilde{U}_p x = \alpha^2 \tilde{U}_p^T H_w H_w^T \tilde{U}_p x.$$

Suppose the matrix $\begin{pmatrix} \tilde{U}_p, & \tilde{U}_p' \end{pmatrix}$ is orthogonal. Then

$$\begin{aligned}
(\tilde{U}_p')^T H_m = 0 &\Rightarrow (\tilde{U}_p')^T S_m \tilde{U}_p' = 0 \\
&\Rightarrow (\tilde{U}_p')^T S_w \tilde{U}_p' + (\tilde{U}_p')^T S_b \tilde{U}_p' = 0 \\
&\Rightarrow (\tilde{U}_p')^T S_w \tilde{U}_p' = 0 \text{ and } (\tilde{U}_p')^T S_b \tilde{U}_p' = 0 \\
&\Rightarrow (\tilde{U}_p')^T H_w = 0 \text{ and } (\tilde{U}_p')^T H_b = 0.
\end{aligned}$$

Here we use (9), the scatter matrix relationship (4), and the fact that S_w and S_b are positive semidefinite. Hence

$$\begin{aligned}
\beta^2 \begin{pmatrix} \tilde{U}_p^T \\ (\tilde{U}_p')^T \end{pmatrix} H_b H_b^T \tilde{U}_p x &= \begin{pmatrix} \beta^2 \tilde{U}_p^T H_b H_b^T \tilde{U}_p x \\ 0 \end{pmatrix} \\
&= \begin{pmatrix} \alpha^2 \tilde{U}_p^T H_w H_w^T \tilde{U}_p x \\ 0 \end{pmatrix} \\
&= \alpha^2 \begin{pmatrix} \tilde{U}_p^T \\ (\tilde{U}_p')^T \end{pmatrix} H_w H_w^T \tilde{U}_p x,
\end{aligned}$$

which implies

$$\beta^2 H_b H_b^T (\tilde{U}_p x) = \alpha^2 H_w H_w^T (\tilde{U}_p x).$$

That is, $\tilde{U}_p x$ and α/β are a generalized singular vector and value of the generalized singular value problem that LDA solves when applied to A. As in the previous section, we need to show that we obtain *all* the LDA solution vectors for A in this way. From

$$S_m^B = \tilde{U}_p^T S_m \tilde{U}_p = \tilde{\Sigma}_p^2, \tag{20}$$

we have that LDA/GSVD applied to B produces $\mathrm{rank}(S_m^B) = p$ non-arbitrary generalized singular value pairs. That is the same number of non-arbitrary pairs as LDA/GSVD applied to A.

We have shown the following.

Theorem 5.1 *If G is an optimal LDA transformation for B, which is the p-dimensional representation of the matrix A via PCA, then $\tilde{U}_p G$ is an optimal LDA transformation for A.*

In other words, LDA applied to A produces

$$Y = (\tilde{U}_p G)^T A = G^T \tilde{U}_p^T A = G^T B,$$

which is the same result as applying PCA to reduce the dimension to p, followed by LDA. Note that if the dimension after the PCA stage is at least $\mathrm{rank}(H_m)$, that is $B = \tilde{U}_l^T A$ for $l \geq p$, the equivalency argument remains unchanged.

An additional consequence of (20) is that

$$\mathrm{null}(S_m^B) = \{0\}.$$

Due to the relationship (4) and the fact that S_w and S_b are positive semidefinite,

$$\mathrm{null}(S_m^B) = \mathrm{null}(S_w^B) \cap \mathrm{null}(S_b^B).$$

Thus the PCA stage eliminates only the joint null space, which is why we don't lose any discriminatory information before applying LDA.

6 QRD Plus LDA

To simplify the computation in the first stage, we note that the same argument holds if we use the reduced QR decomposition [GVL96]

$$A = QR,$$

where $Q \in \mathbb{R}^{m \times n}$ and $R \in \mathbb{R}^{n \times n}$, and let Q play the role that U_q or $\tilde{U}_p$ played before. That is, we use the reduced QR decomposition instead of the SVD.

The n-dimensional representation of A after the QRD stage is

$$B = Q^T A,$$

and the second stage applies LDA to B. Once again letting the superscript B denote matrices after the first stage, we have

$$H_b^B = Q^T H_b \quad \text{and} \quad H_w^B = Q^T H_w.$$

Hence

$$S_b^B = Q^T H_b H_b^T Q \quad \text{and} \quad S_w^B = Q^T H_w H_w^T Q.$$

Suppose

$$S_b^B x = \lambda S_w^B x;$$

i.e. x and λ are an eigenvector-eigenvalue pair of the generalized eigenvalue problem that LDA solves in the second stage. Then, for $\lambda = \alpha^2/\beta^2$,

$$\beta^2 Q^T H_b H_b^T Q x = \alpha^2 Q^T H_w H_w^T Q x.$$

Suppose the matrix $\begin{pmatrix} Q, & Q' \end{pmatrix}$ is orthogonal. Then $(Q')^T A = (Q')^T Q R = 0$, and accordingly, $(Q')^T H_b = 0$ and $(Q')^T H_w = 0$, since the columns of both H_b and H_w are linear combinations of the columns of A. Hence

$$\begin{aligned} \beta^2 \begin{pmatrix} Q^T \\ (Q')^T \end{pmatrix} H_b H_b^T Q x &= \begin{pmatrix} \beta^2 Q^T H_b H_b^T Q x \\ 0 \end{pmatrix} \\ &= \begin{pmatrix} \alpha^2 Q^T H_w H_w^T Q x \\ 0 \end{pmatrix} \\ &= \alpha^2 \begin{pmatrix} Q^T \\ (Q')^T \end{pmatrix} H_w H_w^T Q x, \end{aligned}$$

which implies

$$\beta^2 H_b H_b^T (Qx) = \alpha^2 H_w H_w^T (Qx).$$

That is, Qx and α/β are a generalized singular vector and value of the generalized singular value problem that LDA solves when applied to A. As in the argument for LSI, we show that we obtain *all* the LDA solution vectors for A in this way, by showing that $\mathrm{rank}(S_m^B) = \mathrm{rank}(S_m)$. From the definition (8), we have

$$H_m = A - ce^T = A(I - \frac{1}{n} ee^T) = QR(I - \frac{1}{n} ee^T)$$

and by definition

$$H_m^B = Q^T H_m,$$

and hence

$$H_m = Q H_m^B.$$

Theorem 6.1 *If G is an optimal LDA transformation for B, which is the n-dimensional representation of the matrix A after QRD, then QG is an optimal LDA transformation for A.*

In other words, LDA applied to A produces

$$Y = (QG)^T A = G^T Q^T A = G^T B,$$

which is the same result as applying QRD to reduce the dimension to n, followed by LDA.

7 Experimental Results

7.1 Face databases

In a face database, images of the same person form a cluster, and a new image is recognized by assigning it to the correct cluster. For our experiments, we use facial images from AT&T Laboratories Cambridge and from the Yale face database. The AT&T database[1], formerly the ORL database of faces, consists of ten different images of 40 distinct subjects, for a total of 400 images. This database is widely used by researchers working in the area of face recognition (e.g., [ZCP99], [YY03], and [DY03]). One sample subject or cluster is given in Figure 1(a). Each subject is upright in front of a dark homogeneous background. The size of each image is 92×112 pixels, for a total dimension of 10304. We scaled the images to $46 \times 56 = 2576$ pixels, by averaging the gray levels for 2×2 pixel groups. If we let $A(i, j)$ represent the 8-bit gray level of pixel i in image j, then the data matrix A has 2576 rows and 400 columns.

The Yale face database[2] contains 165 grayscale images of 15 individuals. This database was constructed at the Yale Center for Computational Vision and Control by Belhumeur et al. [BHK97]. There are 11 images per subject, one for each of the following facial expressions or configurations: center-light, with glasses, happy, left-light, without glasses, normal, right-light, sad, sleepy, surprised, and winking. The cluster of images of one individual are shown in Figure 1(b). Compared to the AT&T database, the Yale images have larger variations in facial expression as well as in illumination. Each image is of size 320×243 pixels, for a total dimension of 77760. We scaled these images to 60×45 pixels. If we let $A(i, j)$ represent the 8-bit gray level of pixel i in image j, then the data matrix A has 2700 rows and 165 columns.

7.2 Classification Experiments

We have conducted experiments that compare single-stage to two-stage dimension reduction. In order to cross validate on each database, we leave out one image of each person for testing, and

[1] http://www.uk.research.att.com/facedatabase.html

[2] http://cvc.yale.edu/projects/yalefaces/yalefaces.html

(a) from the AT&T face database

(b) from the Yale face database

Figure 1: A sample cluster

compute trace values, dimension reduction times, and recognition rates as the choice of test image varies.

Tables 2 and 3 confirm our theoretical results regarding traces from Section 3. In the first column of each table, we report traces of the individual scatter matrices in the full space. The next three columns show that these values are preserved by LSI, PCA, and QRD to $\mathrm{rank}(A)$, $\mathrm{rank}(H_m)$, and n, respectively. The last three columns show traces after composing these with LDA/GSVD. In each case, the traces sum to the number of clusters minus one, or $k-1$, matching those shown for single-stage LDA/GSVD and confirming Corollary 2.2.

Figures 2 and 3 illustrate the results of several experiments on the AT&T database. In each graph, we compare classification accuracies in the full space with those after dimension reduction. We also superimpose the pre-processing cost to compute each dimension-reducing transformation in CPU seconds.

The bars report accuracy as a percentage of test images recognized. For classification algorithms [Bis96], we use a centroid-based classification method [PJR03] and K Nearest Neighbor (KNN) classification [TK99]. We measure similarity with the L_2 norm or Euclidean distance. The reduced dimensions after LSI, PCA, and QRD are displayed below the bars on the left, while the final reduced dimension after LDA/GSVD is displayed on the right.

For AT&T, recognition rates are not significantly higher or lower after dimension reduction than in the full space. We observe that LDA/GSVD improves accuracy slightly for test images 1, 3, and 5, lowers it slightly for test images 8 and 9, and leaves it about the same for the rest. In each case, however, recognition rates are more consistent across classifiers with LDA/GSVD. As expected, accuracy is the same when LDA/GSVD is preceded by a first stage of LSI, PCA, or QRD to the appropriate intermediate dimension. The cost of dimension reduction, measured by CPU seconds to compute the transformation for the given training set, is approximately the same for QRD + LDA as for LSI or PCA alone. The advantage of QRD + LDA is that classification of a new image measures similarity of vectors with 39 components instead of about 360 components.

Figures 4 and 5 illustrate the results of our experiments on the Yale database. The graphs clearly show how difficult it is to classify image 4 or image 7 when it's not included in the training set. Each of these images is lit from one side, causing a dark shadow on the opposite side. Despite these variations in illumination, recognition rates are consistently higher after LDA/GSVD than

they were in the full space. In fact, accuracy improves after LDA/GSVD for test images 1, 2, 4, 6, 7, 9, 10, and 11. It degrades for images 5 and 8, and remains consistently perfect for image 3. For each training set, the cost in CPU seconds to compute the dimension reducing transformation is approximately the same for QRD + LDA as for LSI or PCA alone. As for AT&T, QRD + LDA is preferable due to the lower cost of classifying a new image, which involves measuring its similarity to training images in dimension 14 rather than in dimension 140.

8 Conclusion

To address the problem of dimension reduction of very high-dimensional or undersampled data, we have compared four seemingly different methods. Our results are summarized in Table 4, where $q = \mathrm{rank}(A)$, $p = \mathrm{rank}(H_m)$, and the complete orthogonal decomposition is referred to as URV. After showing that both LSI and PCA maximize $J_2(G) = \mathrm{trace}(G^T S_m G)$ over all G with $G^T G = I$, we confirmed the preservation of $\mathrm{trace}(S_w)$ and $\mathrm{trace}(S_b)$ with either method, or with the computationally simpler QRD. The most significant results show the equivalence of the single-stage LDA/GSVD, which extends the applicability of the criterion $J_1(G) = \mathrm{trace}((G^T S_w G)^{-1} G^T S_b G)$ to singular S_w, to any of the two-stage methods. This provides theoretical justification for the increasingly common approach of either LSI + LDA or PCA + LDA, although most studies have reduced the intermediate dimension below that required for equivalence.

Regardless of which of the three approaches is taken in the first stage, LDA/GSVD provides both a method for circumventing the singularity that occurs in the second stage, and a mathematical framework for understanding the singular case. When applied to the reduced representation in the second stage, the solution vectors correspond one-to-one with those obtained using the single-stage LDA/GSVD. Hence the second stage is a straightforward application of LDA/GSVD to a smaller representation of the original data matrix. Given the relative expense of LDA/GSVD and the two-stage methods, we observe that, in general, QRD is a significantly cheaper pre-processing step for LDA/GSVD than either LSI or PCA. However, if $\mathrm{rank}(A) \ll n$, LSI may be cheaper than the reduced QR decomposition, and will avoid the centering of the data required in PCA. In any case, the appropriate two-stage method provides a faster algorithm for LDA/GSVD, without sacrificing its accuracy.

Table 4: Comparison of Two-stage Methods for LDA

Method	LDA/ GSVD	LSI$\rightarrow q$ + LDA/GSVD	PCA$\rightarrow p$ + LDA/GSVD	QRD$\rightarrow n$ + LDA/GSVD
Stage 1		max $\mathrm{trace}(S_m)$	max $\mathrm{trace}(S_m)$	max $\mathrm{trace}(S_m)$
Stage 2	max $\mathrm{tr}(S_w^{-1} S_b)$	max $\mathrm{tr}(S_w^{-1} S_b)$	max $\mathrm{tr}(S_w^{-1} S_b)$	max $\mathrm{tr}(S_w^{-1} S_b)$
cost	URV of $(H_b, H_w)^T$	thin SVD of A	thin SVD of $A - ce^T$	reduced QRD of A

Table 2: Leave-one-out of AT&T database: $\mathrm{trace}(S_w)$ (top) and $\mathrm{trace}(S_b)$ (bottom)

test image	full	LSI	PCA	QRD	LDA/ GSVD	LSI + LDA/ GSVD	PCA + LDA/ GSVD	QRD + LDA/ GSVD
1	1.32×10^4	1.32×10^4	1.32×10^4	1.32×10^4	10^{-27}	10^{-27}	10^{-28}	10^{-28}
	1.94×10^4	1.94×10^4	1.94×10^4	1.94×10^4	39	39	39	39
2	1.32×10^4	1.32×10^4	1.32×10^4	1.32×10^4	10^{-27}	10^{-28}	10^{-28}	10^{-28}
	1.92×10^4	1.92×10^4	1.92×10^4	1.92×10^4	39	39	39	39
3	1.32×10^4	1.32×10^4	1.32×10^4	1.32×10^4	10^{-27}	10^{-28}	10^{-28}	10^{-28}
	1.92×10^4	1.92×10^4	1.92×10^4	1.92×10^4	39	39	39	39
4	1.31×10^4	1.31×10^4	1.31×10^4	1.31×10^4	10^{-27}	10^{-28}	10^{-28}	10^{-28}
	1.92×10^4	1.92×10^4	1.92×10^4	1.92×10^4	39	39	39	39
5	1.31×10^4	1.31×10^4	1.31×10^4	1.31×10^4	10^{-27}	10^{-28}	10^{-28}	10^{-28}
	1.92×10^4	1.92×10^4	1.92×10^4	1.92×10^4	39	39	39	39
6	1.31×10^4	1.31×10^4	1.31×10^4	1.31×10^4	10^{-27}	10^{-28}	10^{-28}	10^{-28}
	1.94×10^4	1.94×10^4	1.94×10^4	1.94×10^4	39	39	39	39
7	1.31×10^4	1.31×10^4	1.31×10^4	1.31×10^4	10^{-27}	10^{-28}	10^{-28}	10^{-28}
	1.94×10^4	1.94×10^4	1.94×10^4	1.94×10^4	39	39	39	39
8	1.31×10^4	1.31×10^4	1.31×10^4	1.31×10^4	10^{-27}	10^{-28}	10^{-28}	10^{-28}
	1.93×10^4	1.93×10^4	1.93×10^4	1.93×10^4	39	39	39	39
9	1.30×10^4	1.30×10^4	1.30×10^4	1.30×10^4	10^{-27}	10^{-28}	10^{-28}	10^{-28}
	1.94×10^4	1.94×10^4	1.94×10^4	1.94×10^4	39	39	39	39
10	1.31×10^4	1.31×10^4	1.31×10^4	1.31×10^4	10^{-27}	10^{-28}	10^{-28}	10^{-28}
	1.94×10^4	1.94×10^4	1.94×10^4	1.94×10^4	39	39	39	39

Table 3: Leave-one-out of Yale database: $\mathrm{trace}(S_w)$ (top) and $\mathrm{trace}(S_b)$ (bottom)

test image	full	LSI	PCA	QRD	LDA/ GSVD	LSI + LDA/ GSVD	PCA + LDA/ GSVD	QRD + LDA/ GSVD
1	1.37×10^4	1.37×10^4	1.37×10^4	1.37×10^4	10^{-28}	10^{-28}	10^{-28}	10^{-28}
	1.44×10^4	1.44×10^4	1.44×10^4	1.44×10^4	14	14	14	14
2	1.38×10^4	1.38×10^4	1.38×10^4	1.38×10^4	10^{-28}	10^{-28}	10^{-28}	10^{-28}
	1.40×10^4	1.40×10^4	1.40×10^4	1.40×10^4	14	14	14	14
3	1.42×10^4	1.42×10^4	1.42×10^4	1.42×10^4	10^{-28}	10^{-28}	10^{-28}	10^{-28}
	1.38×10^4	1.38×10^4	1.38×10^4	1.38×10^4	14	14	14	14
4	9.68×10^3	9.68×10^3	9.68×10^3	9.68×10^3	10^{-28}	10^{-28}	10^{-28}	10^{-28}
	1.54×10^4	1.54×10^4	1.54×10^4	1.54×10^4	14	14	14	14
5	1.41×10^4	1.41×10^4	1.41×10^4	1.41×10^4	10^{-28}	10^{-28}	10^{-28}	10^{-28}
	1.38×10^4	1.38×10^4	1.38×10^4	1.38×10^4	14	14	14	14
6	1.41×10^4	1.41×10^4	1.41×10^4	1.41×10^4	10^{-28}	10^{-28}	10^{-28}	10^{-28}
	1.37×10^4	1.37×10^4	1.37×10^4	1.37×10^4	14	14	14	14
7	1.03×10^4	1.03×10^4	1.03×10^4	1.03×10^4	10^{-28}	10^{-28}	10^{-28}	10^{-28}
	1.50×10^4	1.50×10^4	1.50×10^4	1.50×10^4	14	14	14	14
8	1.41×10^4	1.41×10^4	1.41×10^4	1.41×10^4	10^{-28}	10^{-28}	10^{-28}	10^{-28}
	1.39×10^4	1.39×10^4	1.39×10^4	1.39×10^4	14	14	14	14
9	1.41×10^4	1.41×10^4	1.41×10^4	1.41×10^4	10^{-28}	10^{-28}	10^{-28}	10^{-28}
	1.38×10^4	1.38×10^4	1.38×10^4	1.38×10^4	14	14	14	14
10	1.38×10^4	1.38×10^4	1.38×10^4	1.38×10^4	10^{-28}	10^{-28}	10^{-28}	10^{-28}
	1.39×10^4	1.39×10^4	1.39×10^4	1.39×10^4	14	14	14	14
11	1.41×10^4	1.41×10^4	1.41×10^4	1.41×10^4	10^{-28}	10^{-28}	10^{-28}	10^{-28}
	1.39×10^4	1.39×10^4	1.39×10^4	1.39×10^4	14	14	14	14

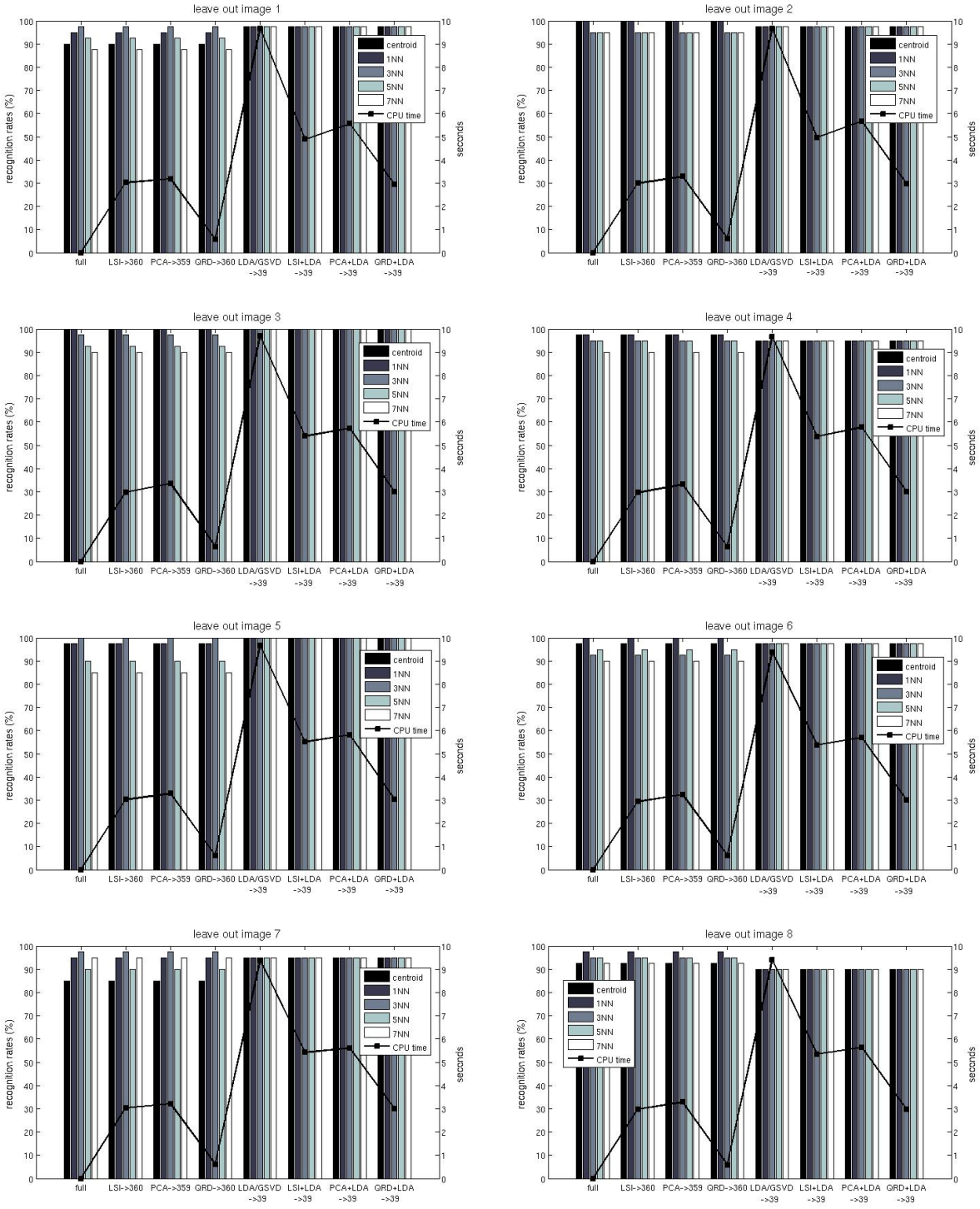

Figure 2: Cross Validation on AT&T database: dimension reduction times and recognition rates

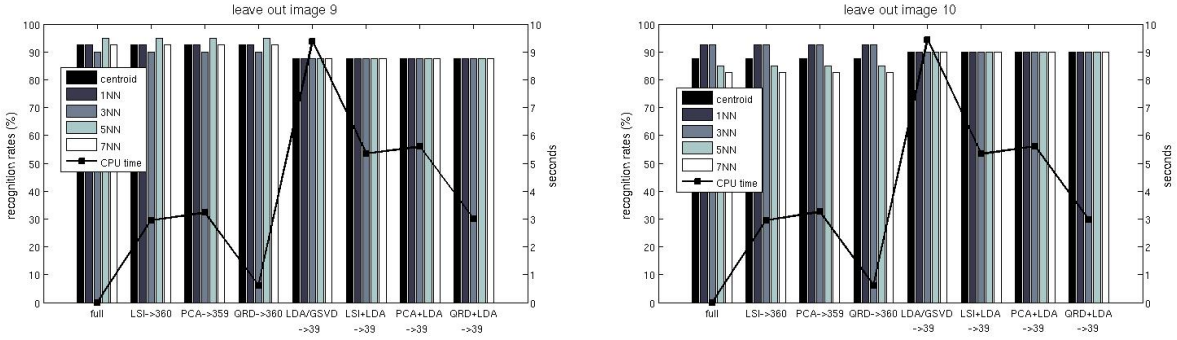

Figure 3: Cross Validation on AT&T database: dimension reduction times and recognition rates

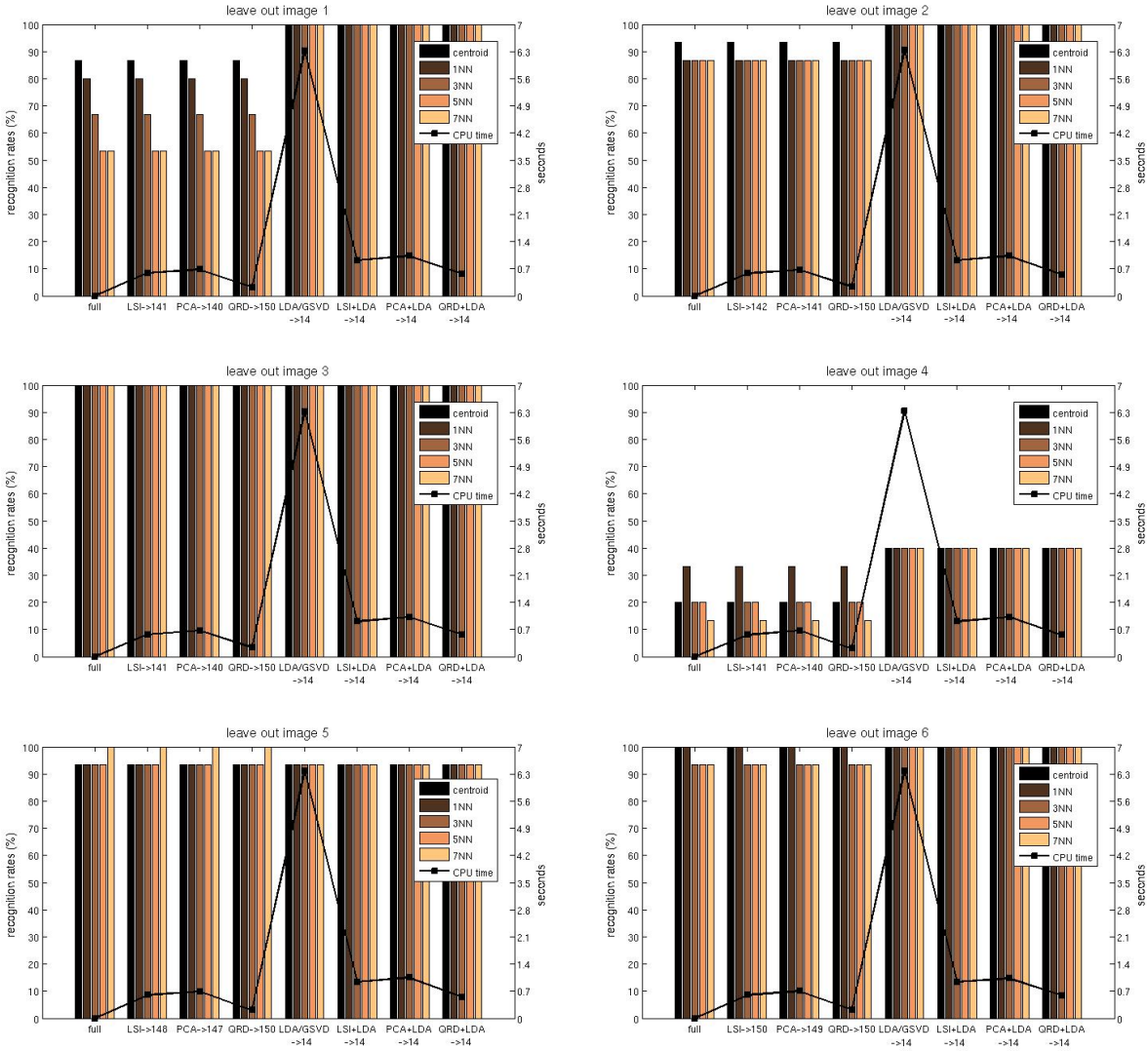

Figure 4: Cross Validation on Yale database: dimension reduction times and recognition rates

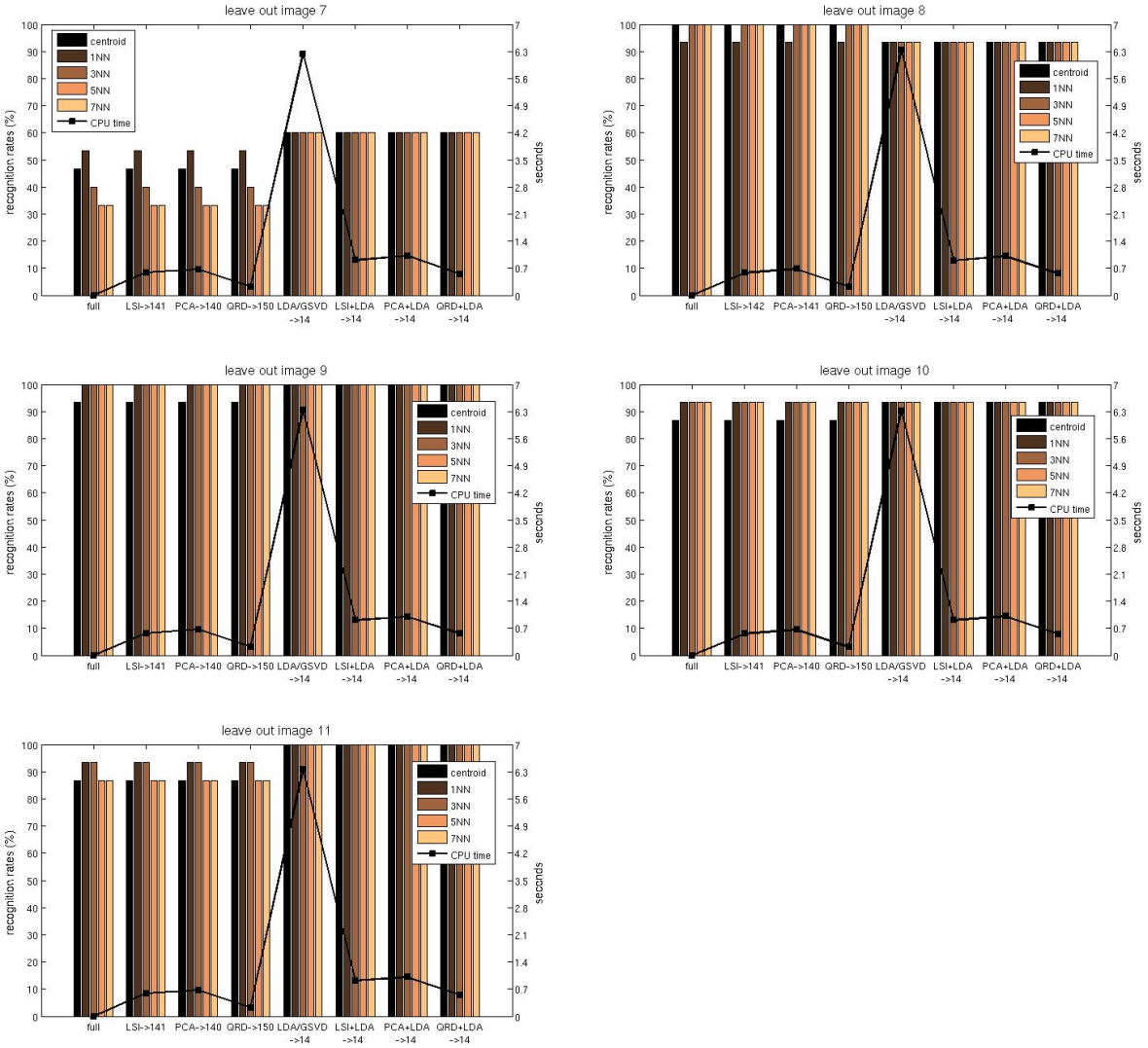

Figure 5: Cross Validation on Yale database: dimension reduction times and recognition rates

References

[BDO95] M.W. Berry, S.T. Dumais, and G.W. O'Brien. Using linear algebra for intelligent information retrieval. SIAM Review, 37(4):573–595, 1995.

[BHK97] P.N. Belhumeur, J.P. Hespanha, and D.J. Kriegman. Eigenfaces vs. fisherfaces: Recognition using class specific linear projection. IEEE Transactions on Pattern Analysis and Machine Intelligence, 19(7):711–720, 1997.

[Bis96] C.M. Bishop. Neural Networks for Pattern Recognition. Oxford University Press, 1996.

[Bjö96] Å. Björck. Numerical Methods for Least Squares Problems. SIAM, 1996.

[DDF+90] S. Deerwester, S. Dumais, G. Furnas, T. Landauer, and R. Harshman. Indexing by latent semantic analysis. J. American Society for Information Science, 41:391–407, 1990.

[DHS01] R. Duda, P. Hart, and D. Stork. Pattern Classification. John Wiley & Sons, Inc., second edition, 2001.

[DY03] D-Q. Dai and P.C. Yuen. Regularized discriminant analysis and its application to face recognition. Pattern Recognition, 36(3):845–847, 2003.

[Fuk90] K. Fukunaga. Introduction to Statistical Pattern Recognition. Academic Press, second edition, 1990.

[GVL96] G.H. Golub and C.F. Van Loan. Matrix Computations. Johns Hopkins University Press, third edition, 1996.

[HJP03] P. Howland, M. Jeon, and H. Park. Structure preserving dimension reduction for clustered text data based on the generalized singular value decomposition. SIAM J. Matrix Anal. Appl., 25(1):165–179, 2003.

[JD88] A.K. Jain and R.C. Dubes. Algorithms for Clustering Data. Prentice Hall, 1988.

[LH95] C.L. Lawson and R.J. Hanson. Solving Least Squares Problems. SIAM, 1995.

[Ort87] J. Ortega. Matrix Theory: A Second Course. Plenum Press, 1987.

[PJR03] H. Park, M. Jeon, and J.B. Rosen. Lower dimensional representation of text data based on centroids and least squares. BIT Numer. Math., 42(2):1–22, 2003.

[PS81] C.C. Paige and M.A. Saunders. Towards a generalized singular value decomposition. SIAM J. Numer. Anal., 18(3):398–405, 1981.

[SW96] D.L. Swets and J. Weng. Using discriminant eigenfeatures for image retrieval. IEEE Transactions on Pattern Analysis and Machine Intelligence, 18(8):831–836, 1996.

[TK99] S. Theodoridis and K. Koutroumbas. Pattern Recognition. Academic Press, 1999.

[Tor01] K. Torkkola. Linear discriminant analysis in document classification. In IEEE ICDM Workshop on Text Mining, 2001.

[VL76] C.F. Van Loan. Generalizing the singular value decomposition. SIAM J. Numer. Anal., 13(1):76–83, 1976.

[YY03] J. Yang and J.Y. Yang. Why can LDA be performed in PCA transformed space? Pattern Recognition, 36(2):563–566, 2003.

[ZCP99] W. Zhao, R. Chellappa, and P.J. Phillips. Subspace linear discriminant analysis for face recognition. Technical Report CS-TR-4009, University of Maryland, 1999.

Algorithm 1 LDA/GSVD

Given a data matrix $A \in \mathbb{R}^{m \times n}$ with k clusters and an input vector $a \in \mathbb{R}^{m \times 1}$, compute the matrix $G \in \mathbb{R}^{m \times (k-1)}$ which preserves the cluster structure in the reduced dimensional space, using

$$J_1(G) = \mathrm{trace}((G^T S_w G)^{-1} G^T S_b G).$$

Also compute the $k-1$ dimensional representation y of a.

1. Compute H_b and H_w from A according to (10) and (7), respectively.
2. Compute the complete orthogonal decomposition

$$P^T K Q = \begin{pmatrix} R & 0 \\ 0 & 0 \end{pmatrix},$$

   where

$$K = \begin{pmatrix} H_b^T \\ H_w^T \end{pmatrix} \in \mathbb{R}^{(k+n) \times m}$$

3. Let $t = \mathrm{rank}(K)$.
4. Compute W from the SVD of $P(1:k, 1:t)$, which is $U^T P(1:k, 1:t) W = \Sigma_b$.
5. Compute the first $k-1$ columns of
   $X = Q \begin{pmatrix} R^{-1} W & 0 \\ 0 & I \end{pmatrix}$, and assign them to G.
6. $y = G^T a$